A class of vector-space bases is introduced for the sparse representation of discretizations of integral operators. An operator with a smooth, non-oscillatory kernel possessing a finite number of singularities in each row or column is represented in these bases as a sparse matrix, to high precision. A method is presented that employs these bases for the numerical solution of second-kind integral equations in time bounded by $O(n \log^2 n)$, where n is the number of points in the discretization. Numerical results are given which demonstrate the effectiveness of the approach, and several generalizations and applications of the method are discussed.

Wavelets for the Fast Solution of Second-Kind Integral Equations

B. Alpert[1,4], G. Beylkin[2,3], R. Coifman[3] , V. Rokhlin[4]

Research Report YALEU/DCS/RR-837
December 1990

[1] Department of Mathematics and Lawrence Berkeley Laboratory, University of California, Berkeley, California 94720
[2] Schlumberger-Doll Research, Ridgefield, CT 06877
[3] Research partially supported by ONR grant N00014-88-K0020
[4] Research partially supported by ONR grants N00014-89-J1527, N00014-86-K0310, and IBM grant P00038486.
Keywords: *Integral Operators, Wavelets, Schultz Method, Numerical Methods*

1 Introduction

Integral equations are a well-known mathematical tool for formulating physical problems. As a numerical tool they have several strengths (good conditioning, dimensionality reduction, and the ability to treat arbitrary regions) but have had one overriding drawback: the high cost of working with the associated dense matrices. For a problem requiring an n-point discretization, the inverse of a dense $n \times n$-matrix must be applied to a vector. Even to apply the matrix itself to a vector requires order $O(n^2)$ operations; application of its inverse by a direct (non-iterative) method requires order $O(n^3)$ operations. If an iterative method is employed, the number of iterations depends on the condition number of the problem and each iteration requires application of the $n \times n$ matrix. For large-scale problems, the resulting costs are often prohibitive.

In recent years a number of algorithms ([5], [10], [11], [14]) have been developed for the fast application of linear operators naturally expressible as dense matrices, the best known of which are the particle simulation algorithms developed by L. Greengard and V. Rokhlin [10]. These schemes combine low-order polynomial interpolation of the function which defines the matrix elements with a divide-and-conquer strategy. They achieve (the equivalent of) order $O(n)$ application of a dense $n \times n$-matrix to a vector.

Over a somewhat longer period, mathematical bases have been constructed with certain *multiscale* properties. Families of functions $h_{a,b}$,

$$h_{a,b}(x) = |a|^{-1/2}\, h\left(\frac{x-b}{a}\right), \qquad a, b \in \mathcal{R},\ a \neq 0,$$

derived from a single function h by dilation and translation, which form a basis for $\mathcal{L}^2(\mathcal{R})$, are known as *wavelets* (Grossman and Morlet [12]). These families have received study by many authors, resulting in constructions with a variety of properties. Meyer [13] constructed orthonormal wavelets for which $h \in C^\infty(\mathcal{R})$. Daubechies [7] constructed compactly supported wavelets with $h \in C^k(\mathcal{R})$ for arbitrary k, and [7] gives an overview and synthesis of the field.

A recent paper [6] establishes a connection between the fast numerical algorithms and the multiscale bases. It introduces the use of wavelets for the application of an integral operator to a function in $O(n \log n)$ operations, where n is the number of points in the discretization of the function. The dissertation [4] of one of the present authors gives an earlier report of the present work. Another paper [2] constructs a class of simple wavelet-like bases for $\mathcal{L}^2[0,1]$ in which a variety of integral operators are sparse. In the present paper, rather than employ a wavelet basis for $\mathcal{L}^2$, we construct a class of bases that transform the dense matrices resulting from the discretization of second-kind integral equations into

sparse matrices. The $n \times n$-matrices resulting from an n-point discretization are transformed to matrices with order $O(n \log n)$ non-zero elements (to arbitrary finite precision). In these bases, the inverse matrices are also sparse, and are obtained in order $O(n \log^2 n)$ operations by a classical iterative method (due to Schulz).

In §2 we present the mathematical construction of the new bases. In §3 we briefly introduce Nyström's method for the solution of integral equations, and show how the wavelet bases result in sparse representation of integral operators and their inverses. We demonstrate that the Schulz method of matrix inversion is efficient in this context. In §4 we present the numerical algorithms for computation of the wavelet bases, transformation of an integral operator into the bases, and computation of its inverse, and we analyze the time complexity of these algorithms. A variety of numerical examples are presented in §5 to demonstrate the effectiveness of the approach, and generalizations and applications are discussed in §6.

2 Wavelet Bases

2.1 Properties of the Bases

Given a set of n distinct points $S = \{x_1, x_2, \ldots, x_n\} \subset \mathcal{R}$ (the discretization) we construct an orthonormal basis for the n-dimensional space of functions defined on S. For simplicity, we assume that $n = k \cdot 2^l$, where k and l are positive integers, and that $x_1 < x_2 < \cdots < x_n$. The basis has two fundamental properties:

1. All but k basis vectors have k vanishing moments, and

2. The basis vectors are non-zero on different scales.

Fig. 1 illustrates a matrix of basis vectors for $n = 128$ and $k = 4$. Each row represents one basis vector, with the dots depicting non-zero elements. The first k basis vectors are non-zero on $x_1, \ldots, x_{2k}$, the next k are non-zero on $x_{2k+1}, \ldots, x_{4k}$, and so forth. In all, one half of the basis vectors are non-zero on $2k$ points from S, one fourth are non-zero on $4k$ points, one eighth are non-zero on $8k$ points, etc. Each of these $n/2 + n/4 + n/8 + \cdots + k = n - k$ basis vectors has k zero moments, *i.e.*, if $b = \langle b_1, \ldots, b_n \rangle$ is one of these vectors, then

$$\sum_{i=1}^{n} b_i \cdot x_i^j = 0, \qquad j = 0, 1, \ldots, k-1.$$

The final k vectors result from orthogonalization of the moments $\langle x_1^j, x_2^j, \ldots, x_n^j \rangle$ for $j = 0, 1, \ldots, k-1$.

These properties of local support and vanishing moments lead to efficient representation of functions which are smooth except at a finite set of singularities. The projection of such a function on an element of this basis will be negligible unless the element is non-zero near one of the singularities. As a simple example, we consider the function $f(x) = \log(x)$ on the interval $[0, 1]$ with the uniform discretization $x_i = i/n$. A hand calculation shows that for any $c > 0$, f may be interpolated on the interval $[c, 2c]$ by a polynomial of degree 7 with error bounded by 4^{-9}, or roughly single precision accuracy. If we choose $k = 8$ in constructing the basis, f will be represented to this accuracy by the k basis vectors non-zero on $x_1, \ldots, x_{2k}$, the k basis vectors non-zero on $x_1, \ldots, x_{4k}$, and so forth, down to the k basis vectors non-zero on $x_1, \ldots, x_n$, in addition to the k orthogonalized moment vectors. The number of non-negligible coefficients in the expansion of f in this basis grows logarithmically in n, the number of points of the discretization. Although this example is idealized, its behavior is representative of the general behavior of an analytic function near a singularity.

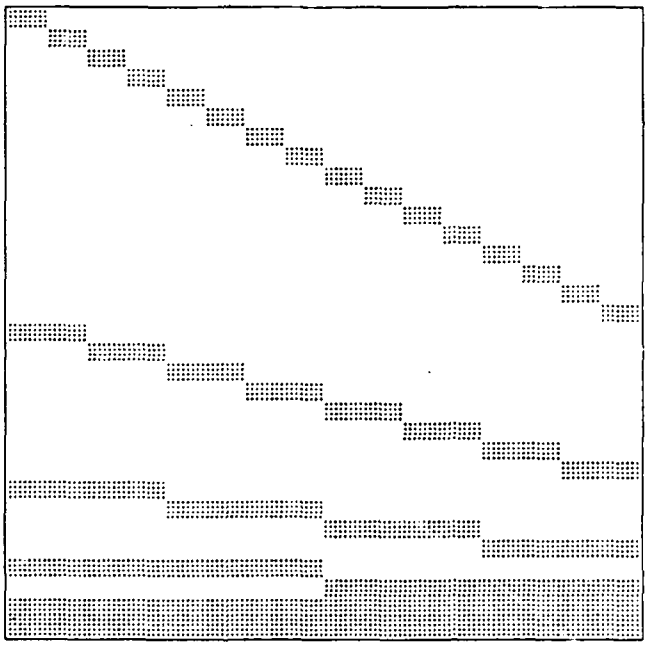

Figure 1: *The matrix represents a wavelet basis for a discretization with* 128 *points, for* $k = 4$. *Each row denotes one basis vector, with the dots depicting non-zero elements. All but the final* k *rows have* k *vanishing moments.*

2.2 Construction of the Bases

The conditions of "local" support and zero moments determine the basis vectors uniquely (up to sign) if we require somewhat more moments to vanish. Namely, out of the k vectors non-zero on $x_1, \ldots, x_{2k}$, we require that one have k vanishing moments, a second have $k+1$, a third have $k+2$, and so forth, and the kth have $2k-1$ vanishing moments. We place the same condition on the k basis vectors non-zero on $x_{2k+1}, \ldots, x_{4k}$, and so on, for each block of k basis vectors among the $n-k$ basis vectors with zero moments.

We construct the basis by construction of a finite sequence of bases (shown in Fig. 2), each obtained by a number of orthogonalizations. The first basis results from $n/(2k)$ Gram-Schmidt orthogonalizations of $2k$ vectors each. In particular, the vectors $\langle {x_1}^j, \ldots, {x_{2k}}^j \rangle$ for $j = 0, \ldots, 2k-1$ are orthogonalized, the vectors $\langle {x_{2k+1}}^j, \ldots, {x_{4k}}^j \rangle$ for $j = 0, \ldots, 2k-1$ are orthogonalized, and so forth, up to the vectors $\langle {x_{n-2k+1}}^j, \ldots, {x_n}^j \rangle$ for $j = 0, \ldots, 2k-1$ which are orthogonalized.

Half of the n vectors of the first basis have at least k zero moments; in forming the second basis, these vectors are retained; the remaining $n/2$ basis vectors are transformed by an orthogonal transformation into basis vectors, each of which is non-zero on $4k$ of the points $x_1, \ldots, x_n$, and half of which have at least k vanishing moments. The orthogonal transformation results from $n/(4k)$ Gram-Schmidt orthogonalizations of $2k$ vectors each. Similarly, the third basis is obtained from the second basis by an orthogonal transformation that itself results from $n/(8k)$ Gram-Schmidt orthogonalizations of $2k$ vectors each. Before we can specify these orthogonalizations, we require some additional notation.

Suppose V is a matrix whose columns $v_1, \ldots, v_{2k}$ are linearly independent. We define W=Orth(V) to be the matrix which results from the column-by-column Gram-Schmidt orthogonalization of V. Namely, denoting the columns of W by $w_1, \ldots, w_{2k}$, we have

$$\begin{array}{c} \text{linear span}\{w_1, \ldots, w_i\} = \text{linear span}\{v_1, \ldots, v_i\} \\ {w_i}^T w_j = \delta_{ij} \end{array} \qquad i, j = 1, \ldots, 2k.$$

For a $2k \times 2k$-matrix V we let V^U and V^L denote two $k \times 2k$-matrices, V^U consisting of the upper k rows and V^L the lower k rows of V,

$$V = \begin{pmatrix} V^U \\ V^L \end{pmatrix}.$$

Now we proceed to the definition of the basis matrices. Given the set of points $S = \{x_1, \ldots, x_n\} \subset \mathcal{R}$ with $x_1 < \cdots < x_n$, where $n = k \cdot 2^j$, we define the $2k \times 2k$

moments matrices $M_{1,i}$ for $i = 1, \ldots, n/(2k)$ by the formula

$$M_{1,i} = \begin{pmatrix} 1 & x_{s_i+1} & \cdots & {x_{s_i+1}}^{2k-1} \\ 1 & x_{s_i+2} & \cdots & {x_{s_i+2}}^{2k-1} \\ \vdots & & & \vdots \\ 1 & x_{s_i+2k} & \cdots & {x_{s_i+2k}}^{2k-1} \end{pmatrix}, \tag{1}$$

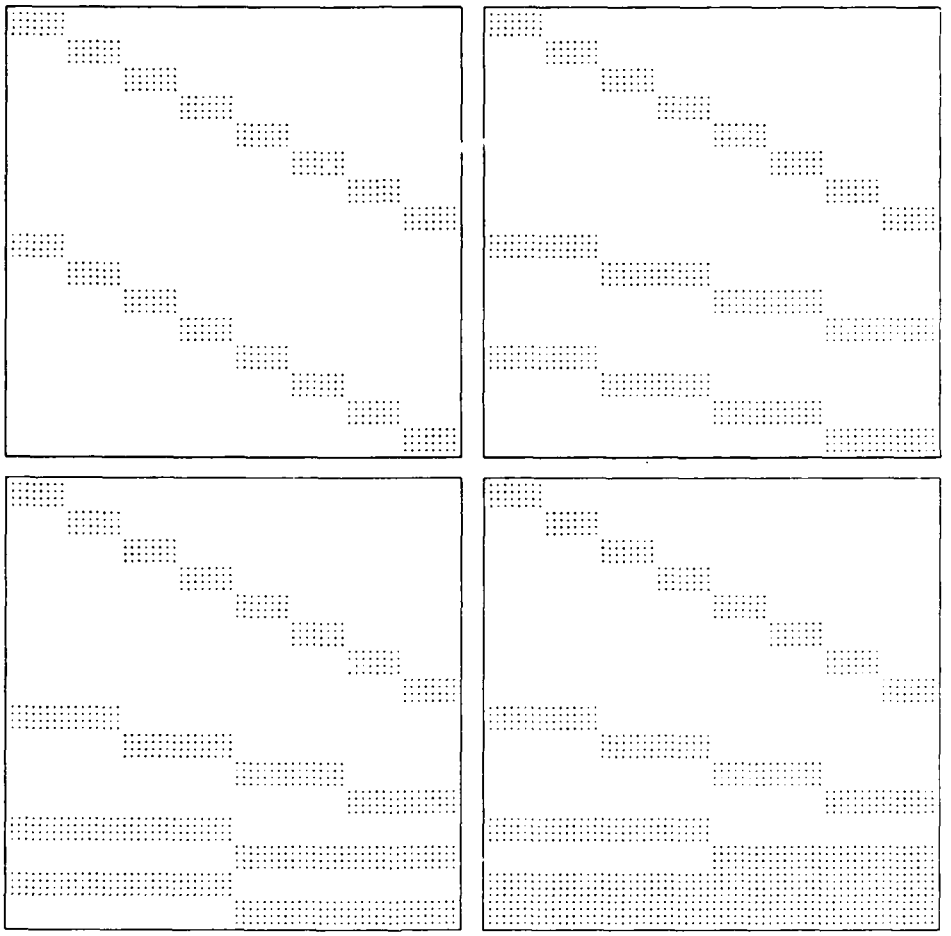

Figure 2: *Each of the four matrices represents one basis, as in Fig. 1. The upper-left matrix is formed by orthogonalizing moment vectors on blocks of $2k$ points. The upper-right matrix is obtained from the upper-left matrix by premultiplying by an orthogonal matrix which is the identity on the upper half. Similarly, the lower matrices are obtained by further orthogonal transformations. The lower-right matrix represents the wavelet basis for $n = 64$, $k = 4$.*

where $s_i = (i-1)2k$. The first basis matrix U_1 is given by the formula

$$U_1 = \begin{pmatrix} {U_{1,1}}^L & & & \\ & {U_{1,2}}^L & & \\ & & \ddots & \\ & & & {U_{1,n_1}}^L \\ {U_{1,1}}^U & & & \\ & {U_{1,2}}^U & & \\ & & \ddots & \\ & & & {U_{1,n_1}}^U \end{pmatrix},$$

where ${U_{1,i}}^T = \mathrm{Orth}(M_{1,i})$ and $n_1 = n/(2k)$. The second basis matrix is $U_2 U_1$, with U_2 defined by the formula

$$U_2 = \begin{pmatrix} I_{n/2} & \\ & U_2' \end{pmatrix}$$

where I_m is the $m \times m$ identity matrix and U_2' is given by the formula

$$U_2' = \begin{pmatrix} {U_{2,1}}^L & & & \\ & {U_{2,2}}^L & & \\ & & \ddots & \\ & & & {U_{2,n_2}}^L \\ {U_{2,1}}^U & & & \\ & {U_{2,2}}^U & & \\ & & \ddots & \\ & & & {U_{2,n_2}}^U \end{pmatrix},$$

where $n_2 = n/(4k)$, ${U_{2,i}}^T = \mathrm{Orth}(M_{2,i})$, and $M_{2,i}$ is given by

$$M_{2,i} = \begin{pmatrix} {U_{1,2i-1}}^U M_{1,2i-1} \\ {U_{1,2i}}^U M_{1,2i} \end{pmatrix}.$$

In general, the jth basis matrix, for $j = 2, \ldots, \log_2(n/k)$, is $U_j \cdots U_1$, with U_j defined by the formula

$$U_j = \begin{pmatrix} I_{n-n/2^{j-1}} & \\ & U_j' \end{pmatrix}$$

where U_j' is given by the formula

$$U_j' = \begin{pmatrix} {U_{j,1}}^L & & & \\ & {U_{j,2}}^L & & \\ & & \ddots & \\ & & & {U_{j,n_j}}^L \\ {U_{j,1}}^U & & & \\ & {U_{j,2}}^U & & \\ & & \ddots & \\ & & & {U_{j,n_j}}^U \end{pmatrix},$$

where $n_j = n/(2^j k)$, $U_{j,i}$ is given by

$$U_{j,i}{}^T = \mathrm{Orth}(M_{j,i}), \tag{2}$$

and $M_{j,i}$ is given by

$$M_{j,i} = \begin{pmatrix} {U_{j-1,2i-1}}^U M_{j-1,2i-1} \\ {U_{j-1,2i}}^U M_{j-1,2i} \end{pmatrix}. \tag{3}$$

The final basis matrix $U = U_l \cdots U_1$, where $l = \log_2(n/k)$, represents the wavelet basis of parameter k on $x_1, \ldots, x_n$.

Remark 2.1 The definitions given for the basis matrices are mathematical definitions only; in a numerical procedure, considerable roundoff error would be introduced by the orthogonalizations defined above. In the actual implementation, the matrices $M_{j,i}$ are shifted and scaled, resulting in a numerically stable procedure that is equivalent to the above definitions (in exact arithmetic). Details of this procedure are provided in §4.

It is apparent that the application of the matrix U to an arbitrary vector of length n may be accomplished in order $O(n)$ operations by the application of $U_1, \ldots, U_l$ in turn. Similarly, $U^{-1} = U^T$ may be applied to a vector in order $O(n)$ operations. Certain dense matrices, in particular those arising from integral operators, are sparse in the basis of U and their similarity transformations can be computed in $O(n \log n)$ operations. These techniques are developed in the following sections.

We conclude this section with an illustration of the vectors of one basis from this class, in Fig. 3.

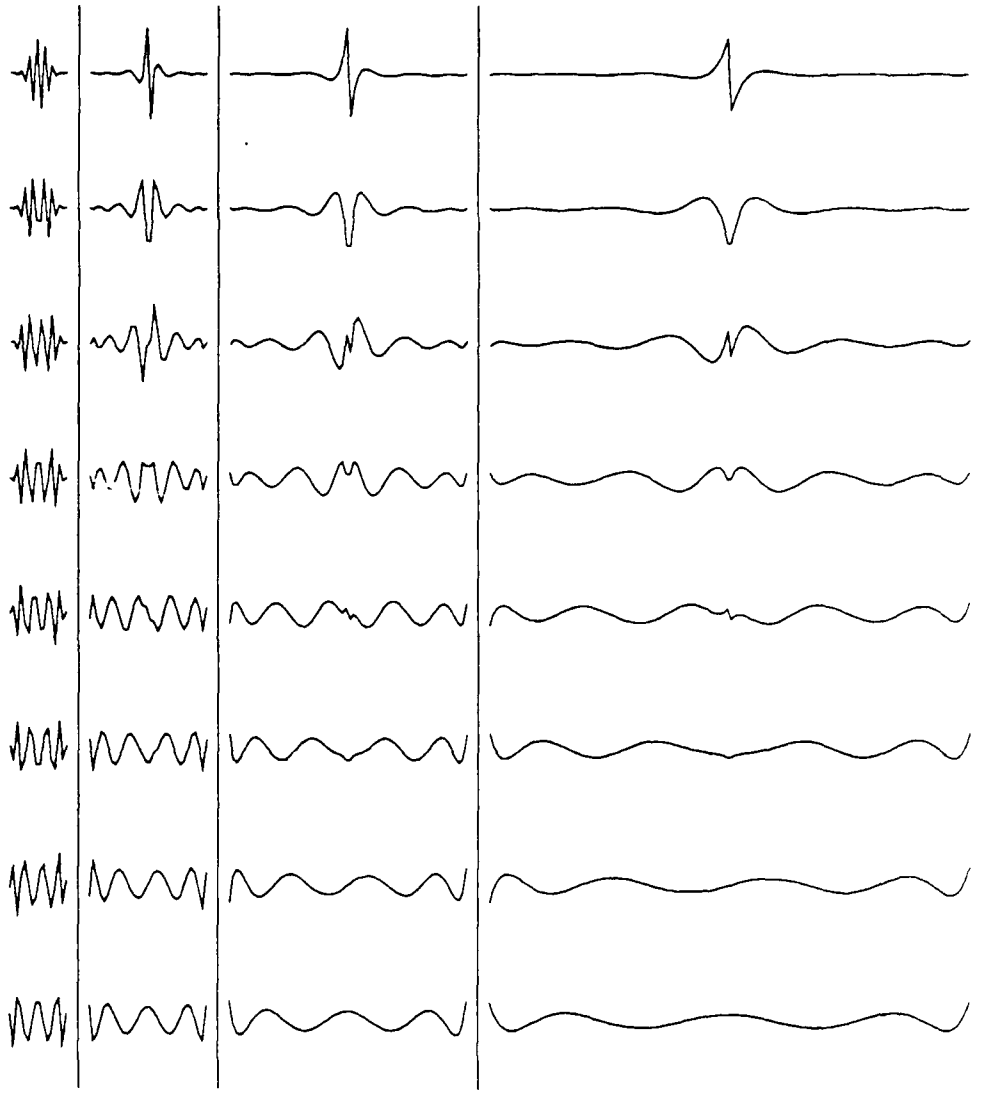

Figure 3: *Basis vectors on four scales are shown for the basis where* $n = 128$, *points* $x_1, \ldots, x_n$ *are equispaced, and* $k = 8$. *The first column of vectors consists of rows* 1–8 *of* U, *the second column consists of rows* 65–72, *etc. Note that half of the vectors are odd and half are even functions, and that the odd ones are generally discontinuous at their center.*

3 Second-Kind Integral Equations

3.1 Nyström Method

A linear Fredholm integral equation of the second kind is an expression of the form

$$f(x) - \int_a^b K(x,t)\, f(t)\, dt = g(x), \tag{4}$$

where the kernel K is in $\mathcal{L}^2[a,b]^2$ and the unknown f and right-hand-side g are in $\mathcal{L}^2[a,b]$. We use the symbol $\mathcal{K}$ to denote the integral operator of Eq. (4), which is given by the formula

$$(\mathcal{K}f)(x) = \int_a^b K(x,t)\, f(t)\, dt,$$

for all $f \in \mathcal{L}^2[a,b]$ and $x \in [a,b]$. Then Eq. (4) written in operator form is

$$(I - \mathcal{K})f = g. \tag{5}$$

The Nyström, or *quadrature*, method for the numerical solution of integral equations approximates the integral operator $\mathcal{K}$ by the finite-dimensional operator R, characterized by points $x_1, x_2, \ldots, x_n \in [a,b]$ and weights $w_1, w_2, \ldots, w_n \in \mathcal{R}$, and given by the formula

$$(Rf)(x) = \sum_{j=1}^{n} w_j\, K(x,x_j)\, f(x_j),$$

for all $f \in \mathcal{L}^2[a,b]$ and $x \in [a,b]$. Substitution of R for $\mathcal{K}$ in Eq. (5), combined with the requirement that the resulting equation hold for $x = x_1, x_2, \ldots, x_n$, yields the following system of n equations in the n unknowns $f_1, f_2, \ldots, f_n$:

$$f_i - \sum_{j=1}^{n} w_j\, K(x_i,x_j)\, f_j = g(x_i), \qquad i = 1, \ldots, n. \tag{6}$$

The approximation $\langle f_1, \ldots, f_n \rangle$ to the solution f of Eq. (4) may be extended to all $x \in [a,b]$ by the natural formula

$$f_R(x) = g(x) + \sum_{i=1}^{n} w_j\, K(x,x_j)\, f_j, \tag{7}$$

which satisfies $f_R(x_i) = f_i$ for $i = 1, \ldots, n$. How large is the error $e_R = f - f_R$ of the approximate solution? We follow the derivation by Delves and Mohamed in [8]. Rewriting Eqs. (6) in operator form, we have

$$(I - R)f_R = g. \tag{8}$$

and combining Eqs. (5) and (8) yields

$$(I - \mathcal{K})e_R = (\mathcal{K} - R)f_R.$$

Provided that $(I - \mathcal{K})^{-1}$ exists, we obtain the error bound

$$\|e_R\| \leq \|(I - \mathcal{K})^{-1}\| \cdot \|(\mathcal{K} - R)f_R\|. \tag{9}$$

The error depends, therefore, on the conditioning of the original integral equation, as is apparent from the term $\|(I - \mathcal{K})^{-1}\|$, and on the fidelity of the quadrature R to the integral operator $\mathcal{K}$. It is not necessary that $\|\mathcal{K} - R\|$ be small, rather merely that R approximate $\mathcal{K}$ well near the solution f. Quadrature rules that have this property, but which are defined only on the points $x_1, \ldots, x_n$, are developed in [3]. In these rules the quadrature weights w_{ij} depend on the point x_i, for $i = 1, \ldots, n$, and the quadrature rules converge rapidly for kernels with diagonal singularities. These rules are used below in the numerical examples of §5.

3.2 Sparse Representation of Integral Operators

We concern ourselves here with kernels $K = K(x,t)$ which are analytic except at $x = t$, where they possess an integrable singularity. We initially discretize the integral operator $\mathcal{K}$ using a simple equispaced quadrature. Given $n \geq 2$, we define points $x_1, \ldots, x_n$ to be equispaced on the interval $[a,b]$,

$$x_i = a + (i-1)(b-a)/(n-1), \tag{10}$$

and define the elements T_{ij} of the $n \times n$-matrix T by the formula

$$T_{ij} = \begin{cases} \frac{1}{n-1} K(x_i, x_j) & i \neq j \\ 0 & i = j. \end{cases} \tag{11}$$

Note that the matrix $T = T(n)$ corresponds to a primitive, trapezoid-like quadrature discretization of the integral operator $\mathcal{K}$. The matrix T possesses the same smoothness properties as the kernel $K(x,t)$. Transformation of T by the bases developed in §2 produces a matrix that is sparse, to high precision. The number of elements is effectively bounded by order $O(n \log n)$.

When the matrix representing the quadrature corrections developed in [3] is added to T, producing high-order convergence to the integral operator, this complexity bound remains valid.

The matrix T, transformed by the orthogonal $n \times n$-matrix U, can be decomposed into the sum of a sparse matrix and a matrix with small norm. Given

$\epsilon > 0$, there exists $c_\epsilon > 0$, independent of n, such that the transformed matrix can be written in the form

$$UTU^T = V + E,$$

where the number of elements in $V = V(n)$ is bounded by $c_\epsilon\, n \log n$ and $E = E(n)$ is small: $\|E\| < \epsilon \|T\|$. We do not prove this assertion here; the proof parallels the proofs of similar statements in [2], but is somewhat more tedious.

3.3 Solution via Schulz Method

The sparse matrix representing the integral operator also has a sparse inverse, which can be computed rapidly.

Schulz's method [15] is an iterative, quadratically convergent algorithm for computing the inverse of a matrix. Its performance is characterized as follows.

Lemma 3.1 *Suppose that A is an invertible matrix, X_0 is the matrix given by $X_0 = A^H/\|A^H A\|$, and for $m = 0, 1, 2, \ldots$ the matrix X_{m+1} is defined by the recursion*

$$X_{m+1} = 2X_m - X_m A X_m.$$

Then X_{m+1} satisfies the formula

$$I - X_{m+1}A = (I - X_m A)^2. \tag{12}$$

Furthermore, $X_m \to A^{-1}$ as $m \to \infty$ and for any $\epsilon > 0$ we have

$$\|I - X_m A\| < \epsilon \quad \text{provided} \quad m \geq 2\log_2 \kappa(A) + \log_2 \log(1/\epsilon), \tag{13}$$

where $\kappa(A) = \|A\| \cdot \|A^{-1}\|$ is the condition number of A and the norm is given by $\|A\|$ = (largest eigenvalue of $A^H A)^{1/2}$.

Proof. Eq. (12) is obtained directly from the definition of X_{m+1}. Bound (13) is equally straightforward. Noting that $A^H A$ is symmetric positive-definite and letting λ_0 denote the smallest and λ_1 the largest eigenvalue of $A^H A$ we have

$$\begin{aligned} \|I - X_0 A\| &= \left\| I - \frac{A^H A}{\|A^H A\|} \right\| \\ &= 1 - \lambda_0/\lambda_1 \\ &= 1 - \kappa(A)^{-2}. \end{aligned} \tag{14}$$

From Eq. (12) we obtain $I - X_m A = (I - X_0 A)^{2^m}$, which in combination with Eq. (14) and simple manipulation yields bound (13). □

The Schulz method is a notably simple scheme for matrix inversion and its convergence is extremely rapid. It is rarely used, however, because it involves

matrix-matrix multiplications on each iteration; for most problem formulations, this process requires order $O(n^3)$ operations for an $n \times n$ matrix. We observe, however, that a sparse matrix, possessing a sparse inverse, whose iterates X_n are also sparse, may be rapidly inverted using the Schulz method. We have seen above that a discretized integral operator $I - T$, similarity-transformed to the representation $A = I - UTU^T$, has only order $O(n \log n)$ elements (to finite precision). In addition, $A^T A$ and $(A^T A)^m$ are similarly sparse. This property enables us to employ the Schulz algorithm to compute A^{-1} in order $O(n \log^2 n)$ operations.

3.4 Oscillatory Coefficients

We now consider a somewhat more general class of integral equations, in which the integral operator is given by the formula

$$(D\mathcal{K}f)(x) = p(x) \int_a^b K(x,t)\, f(t)\, dt,$$

where the kernel K is assumed to be smooth, but the coefficient function p can be oscillatory. In particular, we only restrict p to be positive. In terms of generality, these problems lie between the problems with smooth kernels (and constant coefficient) and those with arbitrary oscillatory kernels.

Writing the corresponding integral equation in operator form, we obtain the equation

$$(I - D\mathcal{K})f = g. \tag{15}$$

Although D is a diagonal operator, and $\mathcal{K}$ is smooth, it is clear that the discretization of the operator $D\mathcal{K}$ will not be a sparse matrix in wavelet coordinates. In this framework, it would appear that the construction of this paper is inapplicable. If we instead consider the operator $D^{1/2}\mathcal{K}D^{1/2}$, in which oscillations in the rows match those in the columns, it becomes clear that the construction of §2 can be revised. Rather than constructing basis functions orthogonal to low-order polynomials x^j, we can construct them to be orthogonal to $p(x)^{1/2}\, x^j$. The sole revision in our definition of basis matrices $U_1, \ldots, U_l$ is to replace the definition (1) of the moments matrices $M_{1,i}$ for $i = 1, \ldots, n/(2k)$, by the new definition

$$M_{1,i} = \begin{pmatrix} p_{s_i+1} & 0 & \cdots & 0 \\ 0 & p_{s_i+2} & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & 0 & p_{s_i+2k} \end{pmatrix} \begin{pmatrix} 1 & x_{s_i+1} & \cdots & {x_{s_i+1}}^{2k-1} \\ 1 & x_{s_i+2} & \cdots & {x_{s_i+2}}^{2k-1} \\ \vdots & & & \vdots \\ 1 & x_{s_i+2k} & \cdots & {x_{s_i+2k}}^{2k-1} \end{pmatrix},$$

where $s_i = (i-1)2k$ and $p_j = p(x_j)^{1/2}$.

Now the integral equation (15) can be transformed to the equation

$$(I - D^{1/2}KD^{1/2})(D^{-1/2}f) = (D^{-1/2}g),$$

which is discretized to a system that is sparse in the revised wavelet coordinates. The inverse matrix is also sparse.

4 Numerical Algorithms

In §2 we defined a class of bases for functions defined on $\{x_1, \ldots, x_n\}$ and in §3 we showed that, to finite precision, second-kind integral operators and their inverses are asymptotically sparse in these bases. In this section we present procedures for computation of the bases, discretized integral operators in these bases, and the inverses of these operators. In §5 we give some numerical examples based on our implementations of these procedures.

The computation of the wavelet bases is discussed next, followed by a discussion of the transformation of the integral operators to the wavelet bases. We defer discussion of the computation of the inverses, sketched above, to §4.3, which contains detailed descriptions of all of the algorithms. Finally, §4.4 gives the complexity analysis for the algorithms.

4.1 Computation of Wavelet Bases

It was mentioned in §2 that the mathematical definition of $U_1, \ldots, U_l$, if used directly, would result in a numerical procedure that would create large roundoff errors. A correct procedure is obtained by shifting and scaling the matrices $M_{j,i}$ defined there.

For a pair of numbers $(\mu, \sigma) \in \mathcal{R} \times (\mathcal{R} \backslash \{0\})$ we define a $2k \times 2k$-matrix $S(\mu, \sigma)$ whose (i, j)th element is the binomial term

$$S(\mu, \sigma)_{i,j} = \binom{j-1}{i-1} \frac{(-\mu)^{j-i}}{\sigma^{j-1}} \tag{16}$$

for $i \leq j$, and $S(\mu, \sigma)_{i,j} = 0$ otherwise. The matrix $S(\mu, \sigma)$ is upper-triangular and non-singular, and its inverse is given by the formula

$$S(\mu, \sigma)^{-1} = S(-\mu/\sigma, 1/\sigma). \tag{17}$$

Furthermore, the product formula

$$S(\mu_1, \sigma_1)S(\mu_2, \sigma_2) = S(\mu_1 + \mu_2\sigma_1, \sigma_1\sigma_2) \tag{18}$$

is easily verified.

We define $M'_{j,i}$ for $j = 1, \ldots, l$ and $i = 1, \ldots, n/(2^j k)$ by the formula

$$M'_{j,i} = M_{j,i} S(\mu_{j,i}, \sigma_{j,i}), \tag{19}$$

where $\mu_{j,i} = (x_{1+(i-1)k2^j} + x_{ik2^j})/2$, $\sigma_{j,i} = (x_{ik2^j} - x_{1+(i-1)k2^j})/2$ and the matrix $M_{j,i}$ is defined by Eq. (1) and Eq. (3) in §2. The matrix $U_{j,i}$ is given by the formula

$$U_{j,i}{}^T = \mathrm{Orth}(M'_{j,i}), \tag{20}$$

which is equivalent to the definition given by Eq. (2). This equivalence immediately follows from the fact that $S(\mu, \sigma)$ is upper-triangular and non-singular.

The matrices $M'_{1,i}$ for $i = 1, \ldots, n/(2k)$ are actually computed by the formula

$$M'_{1,i} = \begin{pmatrix} 1 & \frac{x_{s_i+1}-\mu_{1,i}}{\sigma_{1,i}} & \cdots & \left(\frac{x_{s_i+1}-\mu_{1,i}}{\sigma_{1,i}}\right)^{2k-1} \\ 1 & \frac{x_{s_i+2}-\mu_{1,i}}{\sigma_{1,i}} & \cdots & \left(\frac{x_{s_i+2}-\mu_{1,i}}{\sigma_{1,i}}\right)^{2k-1} \\ \vdots & & & \vdots \\ 1 & \frac{x_{s_i+2k}-\mu_{1,i}}{\sigma_{1,i}} & \cdots & \left(\frac{x_{s_i+2k}-\mu_{1,i}}{\sigma_{1,i}}\right)^{2k-1} \end{pmatrix}, \tag{21}$$

where $s_i = (i-1)2k$. Likewise, the matrices $M'_{j,i}$ for $j = 2, \ldots, l$ and $i = 1, \ldots, n/(2^j k)$ are computed by the formula

$$M'_{j,i} = \begin{pmatrix} U_{j-1,2i-1}{}^U M'_{j-1,2i-1} S^1_{j,i} \\ U_{j-1,2i}{}^U M'_{j-1,2i} S^2_{j,i} \end{pmatrix}, \tag{22}$$

where $S^1_{j,i}$ and $S^2_{j,i}$ are defined by the formulae

$$S^1_{j,i} = S(\mu_{j-1,2i-1}, \sigma_{j-1,2i-1})^{-1} S(\mu_{j,i}, \sigma_{j,i}) \tag{23}$$

$$S^2_{j,i} = S(\mu_{j-1,2i}, \sigma_{j-1,2i})^{-1} S(\mu_{j,i}, \sigma_{j,i}). \tag{24}$$

Application of the inverse and product rules given in Eq. (17) and Eq. (18) to Eq. (23) and Eq. (24) yields formulae by which $S^1_{j,i}$ and $S^2_{j,i}$ can be computed:

$$S^1_{j,i} = S((\mu_{j,i} - \mu_{j-1,2i-1})/\sigma_{j-1,2i-1},\ \sigma_{j,i}/\sigma_{j-1,2i-1}) \tag{25}$$

$$S^2_{j,i} = S((\mu_{j,i} - \mu_{j-1,2i})/\sigma_{j-1,2i},\ \sigma_{j,i}/\sigma_{j-1,2i}). \tag{26}$$

The matrices $M'_{j,i}$ given by Eq. (21) and Eq. (22) are easily seen to be mathematically equivalent to those defined by Eq. (19); nonetheless, computation of $M'_{j,i}$ using Eqs. (21) and (22) avoids the large roundoff errors which would otherwise result.

4.2 Transformation to Wavelet Bases

We assume that for equispaced points $x_1, \ldots, x_n$ (defined in Eq. (10)) and some k the orthogonal matrices $U_1, \ldots, U_l$ defined in §2 have been computed ($l = \log_2(n/k)$). We now present a procedure for computation of UTU^T, where $U = U_l \cdots U_1$ and T is the discretized integral operator defined in Eq. (11).

4.2.1 Simple Example

We begin with a simplified example in which T is replaced by an $n \times n$-matrix V of rank k whose elements V_{ij} are defined by the equation

$$V_{ij} = \sum_{r=1}^{k} \sum_{s=1}^{k} \Lambda_{rs} {x_i}^{r-1} {x_j}^{s-1}, \qquad i,j = 1,\ldots,n.$$

Each row and each column of V contain elements which are the values of a polynomial of degree $k-1$. The matrix V can be written as $V = P^T \Lambda P$, where the elements of the $k \times n$-matrix P are defined by $P_{ij} = {x_j}^{i-1}$ and Λ is the $k \times k$-matrix with elements Λ_{ij}. Recalling that the last k rows of the basis matrix U consist of an orthogonalization of the moment vectors $\langle {x_1}^j, \ldots, {x_n}^j \rangle$ for $j = 0,\ldots,k-1$, we can rewrite V as $V = (P')^T \Lambda' P'$. Here the $k \times n$-matrix P' consists of the last k rows of U and Λ' is a new $k \times k$-matrix with elements Λ'_{ij}.

By the orthogonality of U, it is clear the $n \times n$-matrix $UVU^T = U(P')^T \Lambda' P' U^T$ consists entirely of zero elements except the $k \times k$-submatrix in the lower-right corner, which is the matrix Λ'. Given a function to compute elements of the $n \times n$-matrix V, the matrix Λ' can be computed in time independent of n by using a $k \times k$ extract of values from V. We form the matrix V' with elements V'_{ij} defined by the formula

$$V'_{ij} = V_{in/k,jn/k}, \qquad i,j = 1,\ldots,k. \tag{27}$$

Then $V' = (P'')^T \Lambda' P''$, where P'' is the $k \times k$ extract of P' with elements given by $P''_{ij} = P'_{i,jn/k}$. Thus we obtain

$$\Lambda' = ((P'')^T)^{-1} V' (P'')^{-1} \tag{28}$$

from P'' and V' readily in $O(k^3)$ operations, and we have obtained UVU^T.

4.2.2 General Case

The integral operator matrix T is, of course, not of low rank, but it can be divided into submatrices, each approximately of rank k (see Fig. 4). The submatrices near the main diagonal are of size $k \times k$, those next removed are $2k \times 2k$ and so forth up to the largest submatrices, of size $n/4 \times n/4$. The total number of submatrices is proportional to n/k. Given an error tolerance $\epsilon > 0$, k may be chosen (independently of n) so that each submatrix of T, say T^i, may be written as a sum, $T^i = V^i + E^i$, where the elements of V^i are given by a polynomial of degree $k-1$ and $\|E^i\| < \epsilon \|T^i\|$.

The simplified example, in which the matrix to be transformed is of rank k, is now applicable. Each submatrix of T is treated as a matrix of rank k

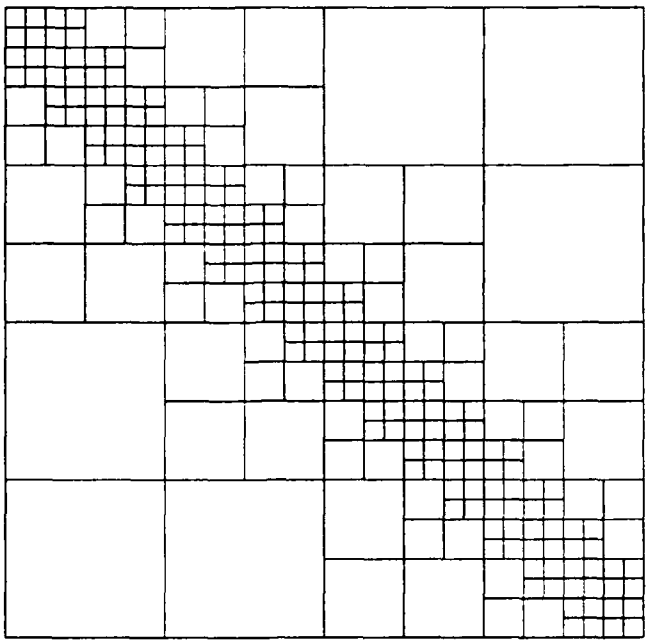

Figure 4: *The matrix represents a discretized integral operator with a kernel that is singular along the diagonal. The matrix is divided into submatrices of rank k (to high precision) and transformed to a sparse matrix with $O(n \log n)$ elements. Here $n/k = 32$.*

and is transformed to wavelet coordinates (for its own scale) in order $O(k^3)$ operations. To make this precise, we write $T = T_0 + \cdots + T_{l-2}$ where T_i consists of the submatrices of size $2^i k \times 2^i k$. For each i, the submatrices of T_i may be interpolated by rank k submatrices, as indicated by the extract of Eq. (27), to obtain matrices V_i. Thus $T_i = V_i + E_i$, where $\|E_i\|$ is small. In the simplified example above, we have shown that the transformed matrices

$$\begin{aligned} W_0 &= V_0 \\ W_1 &= U_1 V_1 {U_1}^T \\ W_2 &= U_2 U_1 V_2 {U_1}^T {U_2}^T \\ &\vdots \\ W_{l-2} &= U_{l-2} \cdots U_1 V_{l-2} {U_1}^T \cdots {U_{l-2}}^T \end{aligned} \tag{29}$$

can be computed by many applications of Eq. (28), all in order $O(nk^2)$ operations. This estimate follows from the fact that there are $O(n/k)$ submatrices, each of which is transformed in $O(k^3)$ operations. Now we define $n \times n$-matrices

$R_0, \ldots, R_l$ recursively:

$$R_i = \begin{cases} W_0 & i = 0 \\ U_i R_{i-1} U_i^T + W_i & i \geq 1 \end{cases} \tag{30}$$

(here $W_{l-1} = W_l = 0$). Then R_l contains the final result, $R_l = U(T - E)U^T$, where $E = E_0 + \cdots + E_{l-2}$.

The matrix-matrix products in the definition of $R_0, \ldots, R_l$ can be computed directly, since the factors and the products contain no more than $O(n \log n)$ elements. A simple implementation with standard sparse matrix structures results in a total operation count of order $O(n \log^2 n)$, but an implementation using somewhat more elaborate data structures, in which repetitive handling of data is avoided, requires only order $O(n \log n)$ operations.

Computation using the result R_l is made more efficient by removing the elements of R_l which can be neglected, within the precision with which R_l approximates UTU^T. For a given precision ϵ, we discard a matrix E' by eliminating elements from R_l below a threshold τ. The threshold depends on the choice of norm; in our implementation, we use the row-sum norm

$$\|A\| = \max_i \sum_{j=1}^{n} |A_{ij}|,$$

for an $n \times n$-matrix A. The element threshold

$$\tau = \frac{\epsilon}{n} \|T\| \tag{31}$$

clearly results in a discarded matrix E' with $\|E'\| < \epsilon \|T\|$.

4.3 Detailed Descriptions of Algorithms

Procedure to compute $U_1, \ldots, U_l$

Comment [Input to this procedure consists of the number of points n, the number of zero moments k, and the points $x_1, \ldots, x_n$. Output is the matrices $U_{j,i}$ for $j = 1, \ldots, l$ and $i = 1, \ldots, n/(2^j k)$, which make up the matrices $U_1, \ldots, U_l$ (note $l = \log_2(n/k)$).]

Step 1.

Compute the shifted and scaled moments matrices $M'_{1,i}$ for $i = 1, \ldots, n/(2k)$ according to Eq. (21).

Step 2.

Compute $U_{1,i}$ from $M'_{1,i}$ by Eq. (20) using Gram-Schmidt orthogonalization for $i = 1, \ldots, n/(2k)$.

Step 3.

Comment [Compute $M'_{j,i}$ and $U_{j,i}$ for $j = 2, \ldots, l$ and $i = 1, \ldots, n/(2^j k)$.]
do $j = 2, \ldots, l$
  do $i = 1, \ldots, n/(2^j k)$
    Compute $U_{j-1,2i-1}{}^U M'_{j-1,2i-1}$ and $U_{j-1,2i}{}^U M'_{j-1,2i}$.
    Compute $S^1_{j,i}$ by Eq. (25) and $S^2_{j,i}$ by Eq. (26);
      multiply to obtain $M'_{j,i}$ by Eq. (22).
    Orthogonalize $M'_{j,i}$ to obtain $U_{j,i}$ by Eq. (20).
  enddo
enddo

Procedure to compute UTU^T

Comment [Input to this procedure consists of n, k, the matrices $U_{j,i}$ computed above, a function to compute elements of T, and the chosen precision ϵ. Output is a matrix R_l such that $\|R_l - UTU^T\| < \epsilon\|T\|$.]

Step 4.

Compute the $k \times k$ extracts, indicated by Eq. (27), of the submatrices of T shown in Fig. 4.

Step 5.

Extract the matrices P'' (Eq. (28)) from U_1, U_2U_1, ..., $U_l \cdots U_1$ and compute $W_0, \ldots, W_{l-2}$ according to Eqs. (29).

Step 6.

Compute $R_0, \ldots, R_l$ by Eq. (30), discarding elements below a threshold τ determined by the precision ϵ (Eq. (31)).

Procedure to compute $UT^{-1}U^T$

Comment [Input to this procedure consists of n, the matrix R_l which approximates UTU^T, and the precision ϵ. Output is a matrix X_m that approximates $UT^{-1}U^T$.]

Step 7.

Compute the matrix $X_0 = R_l^T R_l / \|R_l^T R_l\|$ by direct matrix multiplication, discarding elements below a threshold τ determined by the precision ϵ (Eq. (31)).

Step 8.

Comment [Obtain the inverse by Schulz iteration.]
do $m = 0, 1, \ldots$ **while** $\|I - X_m R_l\| \geq \epsilon$
Compute $X_{m+1} = 2X_m - X_m R_l X_m$, discarding elements below threshold.
enddo

4.4 Complexity Analysis

In the following table, we provide the operation count for each step of the computation of $UT^{-1}U^T$.

Step	Complexity	Explanation
1.	$O(nk)$	There are $n/(2k)$ $2k \times 2k$-matrices; each element of the matrices is computed in constant time.
2.	$O(nk^2)$	For each of the $n/(2k)$ matrices, perform a Gram-Schmidt orthogonalization requiring order $O(k^3)$ operations.
3.	$O(nk^2)$	For each of $n/(4k)+n/(8k)+\cdots+1 = n/(2k)-1$ matrices, compute four products of a $k \times 2k$-matrix with a $2k \times 2k$-matrix, construct two $2k \times 2k$-matrices, and orthogonalize one $2k \times 2k$-matrix.
4.	$O(nk)$	There are $6(1+3+7+\cdots+(n/(2k)-1))+3(n/k)-2$, or order $O(n/k)$, submatrices of T and for each matrix we compute k^2 elements.
5.	$O(nk^2)$	There are $n/(2k)+n/(4k)+\cdots+1 = n/k-1$ matrices P'', each the product of two $k \times k$-matrices. These are each inverted and multiplied with the $O(n/k)$ matrices of the previous step.

Step	Complexity	Explanation
6.	$O(n \log n)$	The diagonally-banded matrix W_0, which contains $O(n)$ elements, grows to $O(n \log n)$ elements by the computation of UW_0U^T, as can be seen by simply examining pictures of W_0 and U. The non-zero elements of the transformed $W_1, \ldots, W_{l-2}$ are a subset of those of W_0.
7.	$O(n \log^2 n)$	Multiplication of two matrices, each with order $O(n \log n)$ elements, to obtain a product with order $O(n \log n)$ elements.
8.	$O(n \log^2 n)$	Two multiplications like that of Step 7 are made per iteration; the number of iterations is independent of n and given by bound (13).
Total	$O(n \log^2 n)$	

5 Numerical Examples

In this section we present operators from several integral equations, the discretization and transformation of the operators to our wavelet bases, and the inversion of the operators via Schulz method.

5.1 Uncorrected Quadratures

We first examine simple quadratures with equal weights, except weight zero at the singularity, as represented by matrix $T = T(n)$ defined by Eq. (11). We transform the matrix $I - T$ to wavelet coordinates as described in §4.2, then compute $(I - T)^{-1}$.

These discretizations are not particularly useful for the solution of the integral equations, due to their slow convergence to the integral operators. They nonetheless make good illustrative examples, for they retain the smoothness of the operator kernels and produce correspondingly sparse matrices. In the next subsection, we examine the results of using high-order quadratures.

For various sizes n of discretization, we tabulate the average number of elements per row in the transformed matrix $U(I-T)U^T$ and the computation time to obtain the matrix. In addition, we display the average number of elements per row of its inverse, and the time to compute the inverse. Finally, we show the error introduced by these computations. The error is determined by the application of the forward and inverse transformations to a random vector: Choose a

vector v of length n with uniformly distributed pseudo-random elements; compute $(I-T)v$ directly, by a standard procedure requiring order $O(n^2)$ operations; transform to wavelet coordinates, obtaining $U(I-T)v$; apply the computed value of $U(I-T)^{-1}U^T$ to the vector $U(I-T)v$; transform to original coordinates by application of U^T; compare the result v' to v. The measure of error is the relative $\mathcal{L}^2$ error, defined by the formula

$$e_{\mathcal{L}^2} = \left(\frac{\sum_{i=0}^{n} [v_i' - v_i]^2}{\sum_{i=0}^{n} {v_i}^2} \right)^{1/2} .$$

The programs to transform and invert, as well as those to determine the error, were implemented in FORTRAN. All computations were performed in double-precision arithmetic on a Sun Sparcstation 1.

The first set of examples is for the kernel $K(x,t) = \log|x-t|$, for a wavelet basis of order $k = 4$ and various choices of precision ϵ. The matrix sparsities, execution times, and errors appear in Table 1. Although the sparse matrices are not banded, we loosely refer to the average number of matrix elements per row as the matrix *bandwidth*. We make the following observations:

1. The bandwidths N_1, N_2 of the operator and its inverse *decrease* with increasing matrix size. In other words, in the range of matrix sizes tabulated, the number of matrix elements grows *sublinearly* in the matrix dimension n.

2. The operator matrix in wavelet coordinates is computed in time that grows nearly linearly in n.

3. The inverse matrix is computed in time which grows sublinearly in n. This is due to the fact that the cost of multiplying the sparse matrices is roughly order $O(nN^2)$, for size n and bandwidth N. One result is that the cost sometimes drops as n increases.

4. The accuracy is within the precision specified. In fact, due to the conservative element thresholding (Eq. 31), the actual error is considerably less than ϵ.

5. The cost increases with increasing precision ϵ, due to the increasing bandwidths generated. The the bandwidths increase approximately as $\log(1/\epsilon)$.

6. For $k = 4$, our fast transformation algorithm does not maintain the specified precision of $\epsilon = 10^{-4}$. This anticipated result follows from the error estimate for polynomial interpolation of logarithm on intervals separated from the origin. An unanticipated attendant result is that the bandwidth increases as the quality of approximation deteriorates (compare to $k = 8$, below). As a result, we did not complete examples for $n = 4096, 8192$.

Table 1: *The integral operator $\mathcal{K}$ defined by the formula $(\mathcal{K}f)(x) = f(x) - \int_0^1 \log|x-t|\, f(t)\, dt$ is discretized, transformed to the wavelet coordinates with $k = 4$, and inverted. For various precisions ϵ and various sizes of discretization, we tabulate the average number of elements/row N_1 of the matrix in wavelet coordinates and the time in seconds t_1 to compute it, corresponding statistics N_2 and t_2 for the inverse, and the error (see text).*

		Transform.		Inversion		$\mathcal{L}^2$
ϵ	n	N_1	t_1	N_2	t_2	Error
10^{-2}	64	7.2	2	8.3	2	0.503E−02
	128	5.9	3	6.5	4	0.257E−02
	256	3.8	7	4.4	4	0.250E−02
	512	2.8	13	3.1	6	0.236E−02
	1024	1.9	26	2.1	6	0.227E−02
	2048	1.4	49	1.4	6	0.221E−02
	4096	1.2	97	1.2	8	0.221E−02
	8192	1.1	195	1.1	12	0.217E−02
10^{-3}	64	17.6	2	19.5	14	0.350E−03
	128	18.1	5	20.0	36	0.270E−03
	256	18.0	11	20.0	83	0.331E−03
	512	14.5	21	15.7	123	0.257E−03
	1024	13.3	41	15.5	262	0.340E−03
	2048	8.5	73	9.8	287	0.233E−03
	4096	5.8	131	6.5	304	0.222E−03
	8192	3.7	242	4.4	312	0.221E−03
10^{-4}	64	28.4	3	30.3	36	0.104E−03
	128	32.1	6	34.3	111	0.140E−03
	256	34.5	15	37.5	302	0.161E−03
	512	33.1	31	35.8	618	0.177E−03
	1024	30.2	63	33.6	1280	0.189E−03
	2048	25.0	121	27.6	2040	0.192E−03

7. The inversion of the 8192×8192 matrix preserving 3-digit accuracy is done in 5 minutes on the Sparcstation. This compares to 95 days (estimated) for inverting the dense matrix by Gauss-Jordan and to 24 minutes for one dense matrix-vector multiplication of that size.

The condition number of the problem, as approximated by the product of the row-sum norms of $U(I - T)U^T$ and its computed inverse, is 3 (independent of size). Five iterations were required by the Schulz method to achieve convergence.

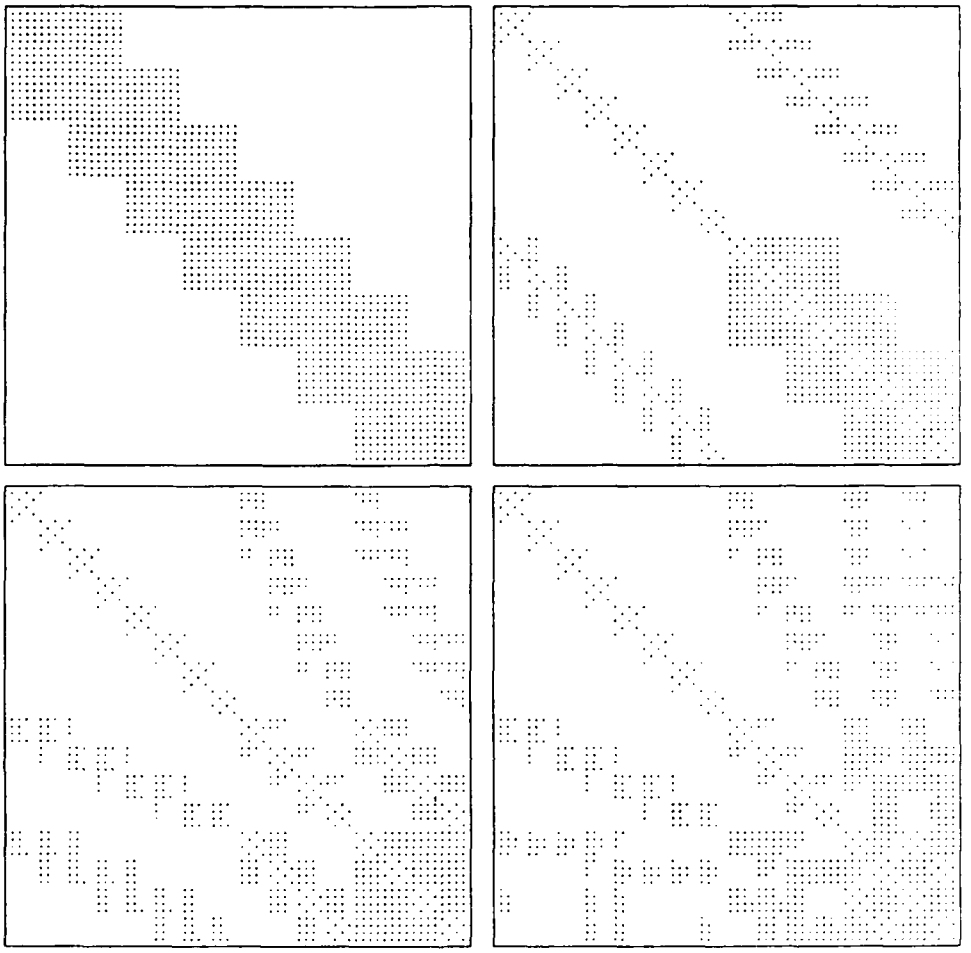

Figure 5: *The matrices constructed in the transformation of $I - T$. matrices $R_0, \ldots, R_3$ defined in Eq. (30), are shown for kernel $K(x,t) = \log|x-t|$. $\epsilon = 10^{-3}$, and $n = 64$. The matrix R_4 looks like R_3 and is not shown.*

In Fig. 5 we show stages in the transformation of the matrix $I - T$. In particular, for $\epsilon = 10^{-3}$ and $n = 64$, the matrices $R_0, \ldots, R_{l-1}$ defined in Eq. (30) are shown. In addition, for $n = 128$ the transformed matrix $U(I-T)U^T$ and its inverse are shown in Fig. 6.

In the next set of examples, for which results are displayed in Table 2, we used the wavelet basis of order $k = 8$. We observe:

1. The bandwidths of the operator matrix and its inverse are less for $k = 8$ than for $k = 4$. The inversion times are correspondingly smaller.

2. The time required to compute the operator matrix is almost four times as large as that for $k = 4$. This is due to the cost of transforming the

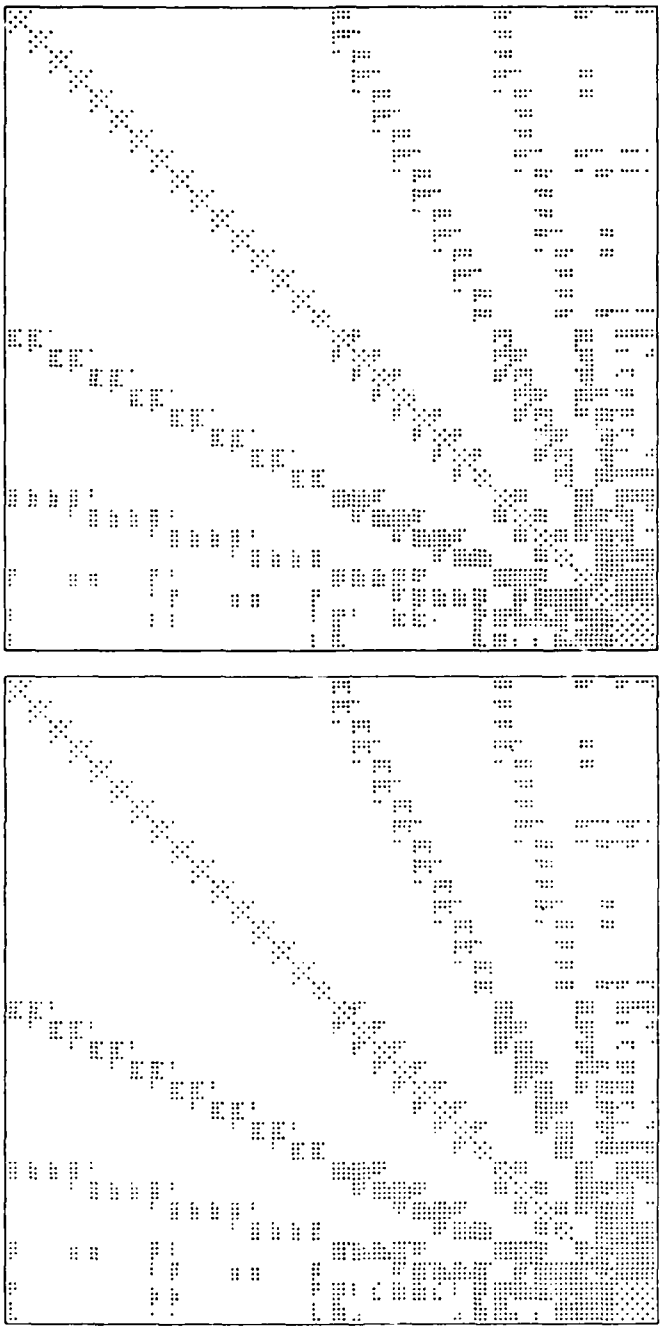

Figure 6: *Transformed matrix $U(I-T)U^T$ (top) and its inverse (bottom) are shown for kernel $K(x,t) = \log|x-t|$, $\epsilon = 10^{-3}$, and $n = 128$.*

near-diagonal band, which is twice as wide for $k = 8$ as for $k = 4$.

3. The obtained accuracy exceeds the specified precision consistently.

Table 2: *The integral operator $\mathcal{K}$ defined by the formula $(\mathcal{K}f)(x) = f(x) - \int_0^1 \log|x-t|\, f(t)\, dt$ is discretized, transformed to the wavelet coordinates with $k = 8$, and inverted. (See Table 1 and text.)*

		Transform.		Inversion		$\mathcal{L}^2$
ϵ	n	N_1	t_1	N_2	t_2	Error
10^{-2}	64	5.8	4	6.2	1	0.191E−02
	128	5.0	10	5.5	2	0.368E−02
	256	3.3	22	3.6	3	0.184E−02
	512	2.7	46	2.9	4	0.113E−02
	1024	1.8	92	1.8	4	0.177E−02
	2048	1.4	182	1.4	5	0.170E−02
	4096	1.2	363	1.3	8	0.928E−03
	8192	1.1	729	1.1	11	0.166E−02
10^{-3}	64	13.4	5	14.5	8	0.373E−03
	128	14.2	13	15.5	21	0.332E−03
	256	13.5	28	14.5	46	0.259E−03
	512	12.7	57	13.6	90	0.225E−03
	1024	10.2	114	11.1	134	0.198E−03
	2048	7.7	221	8.3	176	0.179E−03
	4096	4.9	429	5.2	185	0.174E−03
	8192	3.5	818	3.7	208	0.173E−03
10^{-4}	64	21.8	6	23.0	23	0.280E−04
	128	26.3	15	28.0	81	0.253E−04
	256	28.7	35	31.0	235	0.246E−04
	512	28.4	75	30.9	538	0.184E−04
	1024	25.5	149	27.2	969	0.925E−05
	2048	22.0	297	23.8	1739	0.899E−05
	4096	17.7	561	19.1	2610	0.798E−05

4. As for $k = 4$, the scaling with size n is linear for the transformation step and sublinear for the inversion step.

In the final set of examples in which uncorrected quadratures were used, we perform computations for $k = 4$ and $\epsilon = 10^{-3}$, with various operator kernels. Table 3 presents the results. The first three kernels contain singularities of the types $s(x) = \log(x)$ and $s(x) = x^\alpha$ for $\alpha = \pm\frac{1}{2}$, and are nonsymmetric and non-convolutional. It is readily seen that the bandwidth is strongly dependent on the type of singularity, with the singularity $x^{-1/2}$ producing the greatest bandwidth. We mention also that this particular integral equation is poorly-conditioned:

Table 3: *The integral operator $\mathcal{K}$ defined by the formula $(\mathcal{K}f)(x) = f(x) - \int_0^1 K(x,t)\,f(t)\,dt$, for nonsymmetric, nonconvolutional kernels $K(x,t)$ shown below, is discretized, transformed to the wavelet coordinates with $k = 4$ and $\epsilon = 10^{-3}$, and inverted. (See Table 1 and text.)*

		Transform.		Inversion		$\mathcal{L}^2$
$K(x,t)$	n	N_1	t_1	N_2	t_2	Error
$\cos(xt^2)\log\|x-t\|$	64	18.2	2	20.2	15	0.318E−03
	128	18.6	5	20.4	37	0.302E−03
	256	17.9	11	19.8	82	0.301E−03
	512	14.9	22	16.3	131	0.284E−03
	1024	12.9	42	14.7	242	0.315E−03
	2048	8.5	76	9.5	283	0.241E−03
	4096	5.5	137	6.1	291	0.231E−03
	8192	3.6	252	4.3	310	0.230E−03
$\cos(xt^2)\|x-t\|^{-1/2}$	64	27.2	3	28.9	32	0.256E−03
	128	31.6	7	34.1	122	0.357E−03
	256	35.6	16	40.6	454	0.434E−03
	512	37.3	35	46.3	1509	0.643E−03
	1024	34.5	72	45.4	4166	0.821E−03
$\cos(xt^2)\|x-t\|^{1/2}$	64	6.8	2	7.3	2	0.303E−03
	128	4.4	4	4.7	2	0.204E−03
	256	2.9	8	3.0	3	0.209E−03
	512	2.1	15	2.3	3	0.165E−03
	1024	1.5	30	1.5	3	0.208E−03
	2048	1.4	60	1.4	6	0.909E−03
	4096	1.1	119	1.2	7	0.614E−03
	8192	1.1	242	1.1	12	0.666E−03
$(1+\frac{1}{2}\sin(100x))\times$	64	30.5	3	33.8	44	0.344E−03
$\log\|x-t\|$	128	31.8	6	35.1	103	0.363E−03
	256	21.2	12	24.1	119	0.348E−03
	512	18.6	23	20.7	225	0.372E−03
	1024	15.8	45	18.4	404	0.392E−03
	2048	10.6	82	12.2	466	0.355E−03
	4096	6.4	145	7.4	497	0.336E−03
	8192	4.0	265	4.6	510	0.331E−03

the condition numbers of the discretizations for $n = 64, 128, 256, 512, 1024$ are $9, 17, 34, 98, 469$, respectively.

The fourth kernel provides an example with an oscillatory coefficient $p(x) = (1 + \frac{1}{2}\sin(100x))$. The bases developed in §3.4, which depend on p, are used to transform the discretized integral operator to sparse form. We see in Table 3 that the inverse is also very sparse.

5.2 Solution of Integral Equations

In the preceding subsection, we examined the characteristics of various integral operators and their inverses in wavelet coordinates. We used completely straightforward discretizations; the quadratures represented sums of the integrands at equispaced points (excluding singular points). Such simple quadratures converge too slowly to the integral operators to be of much use in solving integral equations, and we now turn to the high-order quadratures developed in [3].

We first present examples which correspond to the various kernels already tested and shown in Table 3. In Table 4 we tabulate the results, and bandwidth differences from Table 3 reflect the effect of the quadratures.

For the remaining examples we choose integral equations that can be solved analytically, so that the accuracy of the method can be checked. We consider a class of integral equations with logarithmic kernel,

$$f(x) - p(x)\int_0^1 \log|x-t|\, f(t)\, dt = g_m(x), \qquad x \in [0,1], \tag{32}$$

where the right hand side g_m is chosen so that the solution f is given by the formula $f(x) = \sin(mx)$. The integration can be performed explicitly, yielding

$$\begin{aligned}\int_0^1 \log|x-t|\, \sin(mt)\, dt \;=\; & \log(x) - \cos(m)\log(1-x) \\ & - \cos(mx)[\mathrm{Ci}(mx) - \mathrm{Ci}(m(1-x))] \\ & - \sin(mx)[\mathrm{Si}(mx) + \mathrm{Si}(m(1-x))],\end{aligned}$$

where Ci and Si are the cosine integral and sine integral (see, *e.g.*, [1], p. 231). Equation (32) clearly requires quadratures with increasing resolution as m increases; for our examples we let $n = m$, which corresponds to 2π points per oscillation of the right hand side g_m.

Initially we choose coefficient $p(x) = 1$. The results are given in Table 5. Here the error shown is the error of the computed solution relative to the true solution of the integral equation. Many of the observations of the preceding examples can be repeated here; additionally, we make the following comments:

1. The bandwidths are greater than for the uncorrected quadratures, but this effect generally decreases with increasing size.

Table 4: *The integral operator $\mathcal{K}$ defined by the formula $(\mathcal{K}f)(x) = f(x) - \int_0^1 K(x,t)\,f(t)\,dt$, for nonsymmetric, nonconvolutional kernels $K(x,t)$ shown below, is discretized with the corrected trapezoidal rules, transformed to the wavelet coordinates with $k = 4$ and $\epsilon = 10^{-3}$, and inverted. (Compare to Table 3.)*

		Transform.		Inversion		$\mathcal{L}^2$
$K(x,t)$	n	N_1	t_1	N_2	t_2	Error
$\cos(xt^2)\log\|x-t\|$	64	28.3	4	31.6	38	0.164E−03
	128	31.5	9	34.3	103	0.162E−03
	256	30.8	21	33.9	221	0.172E−03
	512	27.0	41	29.7	370	0.177E−03
	1024	21.0	80	23.7	454	0.357E−03
	2048	14.8	143	17.2	566	0.317E−03
	4096	9.5	250	10.4	555	0.282E−03
	8192	5.8	448	6.9	665	0.271E−03
$\cos(xt^2)\|x-t\|^{-1/2}$	64	32.4	4	39.8	87	0.133E−02
	128	38.3	10	45.7	251	0.412E−03
	256	42.7	23	49.3	638	0.464E−03
	512	45.1	51	51.3	1494	0.562E−03
	1024	46.2	110	52.1	3309	0.635E−03
$\cos(xt^2)\|x-t\|^{1/2}$	64	10.4	3	18.4	9	0.867E−03
	128	7.6	6	13.8	13	0.526E−03
	256	5.1	13	9.3	16	0.358E−03
	512	3.3	25	5.2	15	0.292E−03
	1024	2.3	48	3.1	15	0.201E−03
	2048	1.9	96	2.3	20	0.393E−03
	4096	1.5	188	1.7	25	0.405E−03
	8192	1.3	374	1.4	36	0.404E−03

2. The integral equations are solved to within the specified precision in every case but one. The exception, for $\epsilon = 10^{-4}$ and $n = 64$, is likely due to the small number of quadrature points and high specified precision.

3. An integral equation requiring an 8192-point discretization is solved to 3-digit accuracy in less than 20 minutes on the Sparcstation.

For our second set of integral equations, we let the coefficient p be the oscillatory function given by the formula $p(x) = 1 + \frac{1}{2}\sin(100x)$. We carry out the transformation described in §3.4 to solve the integral equation 32. The results are shown in Table 6 and as with Table 5 the error refers to the error of

Table 5: *The integral equations* $f(x) - \int_0^1 \log|x-t|\, f(t)\, dt = g_m(x)$, *for which an explicit solution is known, are solved by the methods of this chapter (compare to Table 1 and see text). For* $\epsilon = 10^{-2}, 10^{-3}, 10^{-4}$ *we set* $k = 4, 4, 8$, *respectively.*

		Transform.		Inversion		$\mathcal{L}^2$
ϵ	n, m	N_1	t_1	N_2	t_2	Error
10^{-2}	64	11.4	3	14.4	7	0.283E−02
	128	10.7	7	13.2	14	0.212E−02
	256	8.6	13	10.6	20	0.140E−02
	512	6.3	26	7.6	26	0.112E−02
	1024	3.6	48	4.5	28	0.821E−03
	2048	1.9	90	2.3	21	0.932E−03
	4096	1.3	174	1.5	15	0.674E−03
	8192	1.1	344	1.1	13	0.499E−03
10^{-3}	64	27.7	4	31.3	36	0.235E−03
	128	31.0	9	34.2	99	0.169E−03
	256	30.6	20	33.6	215	0.161E−03
	512	27.5	41	30.2	377	0.130E−03
	1024	21.7	79	24.4	470	0.597E−03
	2048	15.5	143	18.1	604	0.479E−03
	4096	9.7	248	10.6	579	0.415E−03
	8192	6.0	444	7.3	690	0.354E−03
10^{-4}	64	37.2	8	45.9	78	0.127E−03
	128	47.1	23	56.5	278	0.473E−04
	256	52.9	54	60.9	745	0.311E−04
	512	55.0	118	61.4	1701	0.100E−04
	1024	52.3	248	57.2	3287	0.734E−05

the computed solution relative to the true solution of the integral equation. For the oscillatory coefficient we see performance similar to the constant-coefficient problem, but the cost is higher.

6 Generalizations and Applications

In this paper, we have constructed a new class of vector-space wavelet bases in which a variety of integral operators are represented as sparse matrices. The inverses of these matrices are also sparse, a fact which enables the corresponding integral equations to be solved rapidly. We have asserted that the time complexity

Table 6: *The integral equations* $f(x) - p(x) \int_0^1 \log|x-t|\, f(t)\, dt = g_m(x)$, *for which an explicit solution is known, are solved by the methods of this chapter (compare to Table 1 and see text). For* $\epsilon = 10^{-2}, 10^{-3}, 10^{-4}$ *we set* $k = 4, 4, 8$, *respectively.*

		Transform.		Inversion		$\mathcal{L}^2$
ϵ	n, m	N_1	t_1	N_2	t_2	Error
10^{-2}	64	19.7	4	23.9	18	0.360E−02
	128	17.7	8	21.0	36	0.182E−02
	256	12.6	15	14.6	47	0.174E−02
	512	8.4	29	9.8	57	0.112E−02
	1024	4.7	55	5.7	56	0.104E−02
	2048	2.4	103	2.7	45	0.902E−03
	4096	1.6	198	1.7	38	0.720E−03
	8192	1.3	392	1.3	35	0.543E−03
10^{-3}	64	36.2	4	41.3	63	0.228E−02
	128	40.8	10	47.0	186	0.209E−03
	256	40.5	23	47.3	427	0.177E−03
	512	34.7	46	40.9	712	0.125E−03
	1024	26.6	87	32.5	1042	0.134E−03
	2048	18.7	158	22.5	1065	0.597E−03
	4096	12.2	281	14.2	1127	0.529E−03
	8192	7.2	502	8.4	1104	0.461E−03
10^{-4}	64	47.6	9	58.2	123	0.230E−02
	128	60.7	25	77.3	479	0.180E−03
	256	64.1	59	81.2	1204	0.124E−03
	512	62.5	128	76.3	2492	0.125E−04
	1024	58.8	267	69.3	4672	0.862E−05

for an n-point discretization is bounded by order $O(n \log^2 n)$, but observed order $O(n)$ performance in practice. This cost should be contrasted with a cost of order $O(n^2)$ for direct application of a dense matrix, and order $O(n^3)$ for direct inversion.

A number of limitations exist in the procedures described above. These restrictions may be categorized as "software limitations" and "research questions". We discuss software limitations first.

6.1 Software Limitations

Throughout the paper, we have assumed that the size of the problem n has the form $n = 2^l k$ for some l. This restriction is not fundamental; it merely simplifies the software.

A second software restriction is the assumption of only diagonal singularities. This case is an important one in practice, but in certain situations one may encounter singularities or near-singularities off the main diagonal. The scheme described in §4.2 for transformation of a matrix to wavelet bases can be readily revised to an adaptive scheme, which works as follows: an $m \times m$ submatrix A is transformed to wavelet coordinates, under the assumption that it can be approximated to high precision in each direction by a polynomial of degree less than k. This assumption is then checked by dividing A into four submatrices, each of dimension $m/2 \times m/2$, transforming each submatrix, and "glueing" the pieces together. If the results from the two computations match (to high precision), no further refinement of the original submatrix is needed. Otherwise, the procedure is repeated recursively on the $m/2 \times m/2$ submatrices. The cost of this adaptive procedure is roughly 5 times as great as the cost of a static procedure in which the structure of the singularities is known *a priori*.

6.2 Research Questions

The list of research issues is of course much longer. One of the most pressing issues is the generalization to two and three dimensions. Although conceptually the generalization of the wavelet bases to several dimensions is quite straightforward (see, *e.g.*, [2]), actual procedures to perform the required orthogonalizations have not been developed. Also, the issue of high-order quadratures for two and three dimensions has not been resolved.

Another question is whether similar "custom-constructed" bases can be used to create sparse representations of integral operators with oscillatory kernels. Initial efforts in this direction for a limited class of such operators, in particular for Fourier transforms with non-equispaced points and frequencies, appear promising [9].

6.3 Applications

In this paper the primary application of our new wavelet bases has been the solution of second-kind integral equations. The bases are very effective for the fast solution of a wide class of such problems. In addition, many other classes of problems can be solved efficiently using these techniques. We list a few of these problem types.

1. Elliptic partial differential equations rewritten as integral equations by the Lippman-Schwinger method, in which the Green's functions are non-oscillatory.

2. Evolution of homogeneous parabolic PDEs with constant or periodic boundary conditions, by explicit time steps. This method consists of repeated squarings of the operator for a single time step, leading to an order $O(n \log t)$ algorithm for evolving an n-point discretization for t time steps.

3. Evolution of general parabolic PDEs by implicit time steps, in which the elliptic problem on each time step is solved in wavelet coordinates.

4. Evolution of hyperbolic PDEs by a method of operator squaring analogous to the scheme proposed for homogeneous parabolic PDEs above.

5. Problems of potential theory and pseudo-differential operators.

6. Signal compression, including signals of seismic, visual, and vocal origin. There is also reason to expect that analysis of such compressed data will be simpler than analysis of data resulting from less efficient compression schemes.

In this paper we strayed from the original mathematical definition of wavelets to construct classes of bases tailored for numerical computation. The basis vectors' principal properties of local support and vanishing moments lead to sparse representations of functions and operators that are smooth except at a small number of singularities. There is little doubt that other bases can be constructed along similar lines to possess various properties. One current challenge is the construction of bases suitable for the efficient representation of a variety of oscillatory operators.

References

[1] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions.* National Bureau of Standards, Washington, DC, 1972.

[2] B. Alpert. A class of bases in L^2 for the sparse representation of integral operators. Technical report, Lawrence Berkeley Laboratory, University of California, Berkeley, CA, 1990.

[3] B. Alpert. Rapidly-convergent quadratures for integral operators with singular kernels. Technical report, Lawrence Berkeley Laboratory, University of California, Berkeley, CA, 1990.

[4] B. Alpert. *Sparse Representation of Smooth Linear Operators.* PhD thesis, Yale University, December, 1990.

[5] B. Alpert and V. Rokhlin. A fast algorithm for the evaluation of Legendre expansions. Technical report, Department of Computer Science, Yale University. To appear, *SIAM Journal on Scientific and Statistical Computation*, 1991.

[6] G. Beylkin, R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms I. Technical report, Department of Computer Science, Yale University. To appear, *Communications in Pure and Applied Mathematics*, 1990.

[7] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, XLI:909–996, 1988.

[8] L. M. Delves and J. L. Mohamed. *Computational Methods for Integral Equations.* Cambridge University Press, 1985.

[9] A. Dutt and V. Rokhlin. Personal communication.

[10] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325–348, 1987.

[11] L. Greengard and J. Strain. The fast Gauss transform. Technical report, Department of Computer Science, Yale University, 1989.

[12] A. Grossman and J. Morlet. Decomposition of Hardy functions into square integrable wavelets of constant shape. *SIAM Journal on Mathematical Analysis*, 15:723–736, 1984.

[13] Y. Meyer. Principe d'incertitude, bases Hilbertiennes et algèbres d'opérateurs. Technical report, Séminaire Bourbaki, 1985-1986, nr. 662.

[14] S. O'Donnell and V. Rokhlin. A fast algorithm for the numerical evaluation of conformal mappings. *SIAM Journal on Scientific and Statistical Computing*, 10:475–487, 1989.

[15] G. Schulz. Iterative berechnung der reziproken matrix. *Zeitschrift für Angewandte Mathematik und Mechanik*, 13:57–59, 1933.